Mikko Salminen

GSM/GPRS -MODUULIN ETÄHALLINTASOVELLUS

Tietotekniikan koulutusohjelma
Tietoliikennetekniikan suuntautumisvaihtoehto
2009

GSM/GPRS -MODUULIN ETÄHALLINTASOVELLUS

Salminen, Mikko
Satakunnan ammattikorkeakoulu
Tietotekniikan koulutusohjelma
Toukokuu 2009
Aarinen, Reino
UDK: 527.62, 621.39
Sivumäärä: 41

Asiasanat: etäkäyttö, GPRS, satelliittipaikannus, PHP

Tämän opinnäytetyön aiheena oli GSM/GPRS -moduulin etähallinta ja siihen tehtävä sovellus. Monipuoliseen käyttöön soveltuvan ja GPS-antennin sisältävän moduulin etähallinta toteutettiin PHP-pohjaisella internet-sovelluksella yhdessä MySQL-tietokannan kanssa. Sovelluksen avulla mahdollistettiin moduulin ymmärtämien AT-komentojen lähettäminen moduulille sen GPRS-yhteyttä hyväksi käyttäen.

Työssä pyrittiin myös panostamaan sovelluksen tietoturvallisuuteen ja helppokäyttöisyyteen. Laitetta voitiin konfiguroida sisältä Python-koodin avulla, jossa määriteltiin perusasetuksia tiedonsiirtoa varten.

Osana opinnäytetyötä toteutettiin lisäksi etähallintaa laajentava karttasovellus, jossa moduulilta saatu sijainti pystyttiin näyttämään dynaamisella kartalla. Työssä tutustuttiin lisäksi sovelluksen pohjalla toimivien tekniikoiden teoriaan ja historiaan.

REMOTE CONTROL APPLICATION FOR A GSM/GPRS -MODULE

Salminen, Mikko
Satakunnan ammattikorkeakoulu, Satakunta University of Applied Sciences
Degree Programme in Information Technology
May 2009
Aarinen, Reino
UDC: 527.62, 621.39
Number of pages: 41

Key words: remote control, GPRS, GPS, PHP

The subject of this thesis was the remote control of a GSM/GPRS module and an application made for it. The module includes a GPS-antenna and is suitable for diverse use. The remote control of the module was implemented via a PHP-based Internet application along with a MySQL database. Using GPRS connectivity, the application is able to send AT-commands to the module.

In the thesis, the purpose was to invest in the safety and user friendliness of the application. The module was able to be configured from the inside with a Python code, in which the basic settings for data transfer were specified.

As part of this thesis, an expansive map application was made. The application receives the position information from the module and shows it on a dynamic map. In addition, theory and history of the techniques behind the application were explored.

SISÄLLYS

1 JOHDANTO

Työssä perehdytään raumalaisen merielektroniikkayrityksen Rauma Electronics Oy:n tilaamaan GSM/GPRS -moduulin etähallintaan ja itse moduuliin. Laitteessa on tavallinen, suomalaisen operaattorin SIM-kortti ja GPRS-toiminnallisuus sekä lisäksi GPS-antenni.

Tavoitteena on tehdä PHP-pohjainen etähallintasivusto, jonka avulla moduulia pystytään hallinnoimaan internet-sivun kautta. Lisäksi tehdään hallintasivun laajennuksena erillinen karttasovellus, jossa saadaan haettua laitteen sijainti sivulle upotettuun, dynaamiseen karttaan. Moduulia ohjataan erityisillä AT-komennoilla, joihin koko hallinta perustuu.

Sovellus tulee aluksi vain Rauma Electronics Oy:n käyttöön, mutta se on mahdollista ottaa käyttöön myös laajemmin. Moduulin ja sen etähallinnan käyttömahdollisuuksia on rajaton määrä, ja tämä sovellus toimii tulevaisuudessa tarvittavien toimintojen perustana, jonka päälle voidaan rakentaa lähes mitä tahansa.

2 TEKNIIKAT SOVELLUKSEN TAUSTALLA

2.1 PHP

PHP on tanskalais-grönlantilaisen Rasmus Lerdorfin luoma ja The PHP Groupin kehittämä, laajasti käytetty, alustasta riippumaton web-ohjelmointikieli, jota käytetään erityisesti dynaamisten web-sivujen luonnissa. PHP on komentosarjakieli, jossa ohjelmakoodi tulkitaan vasta ohjelman suoritusvaiheessa. Lyhenne PHP tuli alun perin sanoista Personal Home Page, mutta se muutettiin myöhemmin merkitsemään sanoja PHP: Hypertext Preprocessor. /1/

PHP on suhteellisen yksinkertainen kieli, jolla on todella laaja luokkakirjasto eri toiminta-alueiden käsittelyyn sekä tuki useimmille tietokannoille. Nämä ominaisuudet tekevät PHP:stä hyvän ja suositun kielen web-ohjelmointiin, ja nykyisin PHP onkin johtava dynaamisten web-palveluiden tuottamiseen tarkoitettu ohjelmointikieli /2/. PHP:tä hyödyntävät mm. tunnetut yhteisösivustot Facebook ja IRC-Galleria. /1/

2.1.1 Syntaksi

PHP:tä käytetään yleisimmin upotettuna HTML-sivujen sisälle PHP:n aloitus- ja lopetustagien ”<?php” ja ”?>” väliin. Kun selain lähettää pyynnön PHP-sivusta, palvelin käyttää koodin PHP-tulkilla. Tulkki käsittelee vain tekstin, joka on PHP-tagien sisällä ja palauttaa valmiin sivun palvelimelle, joka lähettää sen edelleen selaimelle. /1/

```
<html>
 <head>
  <title>Yksinkertainen esimerkki</title>
 </head>
 <body>
  <?php echo '<h1>Näin PHP toimii!</h1>'; ?>
 </body>
</html>
```

Edellinen PHP-sivu palautuu siis selaimelle seuraavassa muodossa:

```
<html>
<head>
 <title>Yksinkertainen esimerkki</title>
</head>
 <body>
  <h1>Näin PHP toimii!</h1>
 </body>
</html>
```

2.1.2 Muuttujat

PHP:ssä muuttujat ovat heikosti tyypitettyjä, eli ne voivat saada minkä tyyppisen arvon tahansa riippuen kontekstista. Muuttujat merkitään PHP:ssä dollarimerkillä nimen alussa. Ne voivat sisältää kirjaimet a-z, numeroita sekä alaviivoja, mutta muuttujan nimi ei saa alkaa numerolla. Muuttuja voi olla esimerkiksi ”$muuttujan_nimi”. PHP:n muuttujatyypit ovat totuusarvo (boolean), kokonaisluku (integer), liukuluku (float), merkkijono (string), joukko (array), olio (object), ulkoinen resurssi (resource) ja tyhjä (NULL) /3/. /1/

2.2 MySQL

MySQL on ilmainen, alustasta riippumaton, tehokas ja varsinkin web-palveluiden yhteydessä suosittu SQL-tietokannan hallintajärjestelmä. MySQL:llä on suomalainen tausta, sillä tietokantaa oli luomassa suomalainen Michael Widenius yhdessä ruotsalaisen David Axmarkin kanssa. Ensimmäinen versio julkaistiin vuonna 1996 ja nykyisin sitä kehittää ruotsalainen MySQL AB, jonka Sun Microsystems osti 2008. /4/

MySQL sisältää rajapinnan useille ohjelmointikielille, mutta useimmiten sitä käytetään tietokantana ns. LAMP-alustaisissa web-palveluissa, joissa pohjana on Linux-käyttöjärjestelmä, Apachen-palvelin, MySQL-tietokanta sekä PHP-, Python- tai Perl-ohjelmointikieli. MySQL:n hallinnointi tapahtuu helposti komentoriviltä, tekstipohjaisella asiakasohjelmalla tai graafisilla työkaluilla, kuten MySQL Administrator, MySQL Query Browser tai phpMyAdmin. MySQL:ää hyödyntäviä tunnettuja tahoja ovat mm. Wikipedia, Google ja Yahoo. /4/

2.3 Python

Python on suhteellisen uusi, yksinkertainen, monipuolinen ja tulkattava ohjelmointikieli, joka yhdistää useiden kielten parhaita ominaisuuksia yhdeksi toimivaksi kokonaisuudeksi. Python eroaa perinteisistä ohjelmointikielistä monellakin tavalla. Python-ohjelmaa ei tarvitse kääntää ennen ohjelman suoritusta, vaan suoritusympäristö kääntää sen lennossa Javan tavukoodin kaltaiselle välikielelle, jota suoritusympäristö sitten suorittaa. Python on myös ns. heikosti tyypitetty kieli, eli kielessä esiintyville muuttujille ei määritellä lainkaan tietotyyppejä, vaan jokainen muuttuja on tyypittämätön viite, joka voi osoittaa millaiseen olioon tahansa. Pythonin perusfilosofia onkin säästää ohjelmoijan aikaa tietokoneen ajan kustannuksella. /5, s.253–255/

2.4 Salaus

PHP-sivuilla pyritään usein estämään asiattomien käyttäjien pääsy sivuille käyttämällä kirjautumista. Salausta käytetään useimmiten vain kirjautumisen yhteydessä annettavien tunnusten ja salasanojen salaamiseen, jos koko yhteyttä ei tarvitse salata. Näiden komponenttien salaamiseen on käytettävissä useita eri algoritmeja, joista PHP sisältää valmiina funktioina MD5:n, SHA1:n ja CRC32:n, joista SHA1 on tosin käytössä vain versiosta 4.3.0 eteenpäin /6, 7, 8/.

2.4.1 MD5

MD5 (Message Digest 5) on Ronald Rivestin kehittämä tiivistealgoritmi, jota käytetään yleisesti viestin salaukseen keskitason salausta vaativissa sovelluksissa. MD5 pohjautuu vahvasti edeltäjäänsä MD4:ään, mutta korkeamman turvallisuuden saavuttaakseen siitä tuli jonkin verran MD4:ää hitaampi. MD4:ää lähdettiin kehittämään paremmaksi versioksi, kun analyysit osoittivat sen mahdollisesti epäturvalliseksi. Uusi versio MD5 julkaistiin keväällä 1992 ja se sisälsi useita parannuksia, kuten neljännen laskentakierroksen. /9, 10/

MD5-algoritmi tuottaa tuloksenaan 128-bittisen tiivisteen. Tyypillisesti tiiviste esitetään heksakoodatussa muodossa, jolloin se on 32 merkkiä pitkä. Algoritmin idea ly-

hykäisyydessään on se, että sen tuottamasta tiivisteestä ei pysty päättelemään mitään salattavan viestin sisällöstä. Jo yhden merkin ero lähdetekstissä saa aikaan täysin erilaisen tiivisteen. Matemaattisesti sellaisia merkkijonoja, jotka tuottavat saman tiivisteen on rajattomasti. Niiden löytäminen on kuitenkin erittäin työlästä ja jopa epärealistista. /9, 10/

MD5-tiivisteestä on mahdotonta tuottaa käänteisesti alkuperäisen viestin sisältö ja siksi sitä kutsutaankin yhden suunnan hajakoodausalgoritmiksi. Salauksen murtamiseen onkin käytettävä eri lähestymistapaa. Voidaan yrittää itse luoda MD5-tiivisteitä eri sanoista, merkeistä ja niiden yhdistelmistä ja verrata niitä alkuperäiseen. Jos tulos on sama, on lähes varmasti myös lähdeteksti sama. Tavallisten kotitietokoneiden tehot eivät yleensä kuitenkaan riitä tällaisen menettelyn tehokkaaseen käyttöön. /10/

MD5 onkin peruskäytössä toimiva ja turvallinen algoritmi, mutta kohtalaisen murrettavuuden takia korkeampaa tietoturvaa vaativat tahot käyttävät yleensä esimerkiksi SHA-1 -tiivisteitä, jotka ovat 160-bittisiä. MD5-algoritmia käytetään laajasti mm. Unix-järjestelmien salasanakoodauksessa ja avoimen lähdekoodin ohjelmissa myös tarkistesummana muuttumattoman ohjelmakoodin varmistamiseen. /10/

2.5 Google Maps API

Google Maps API on Googlen suositun karttasovelluksen web-sivuille upotettava ohjelmointirajapinta, joka mahdollistaa karttojen muokkaamisen ja käytön omiin tarpeisiin ja jonka pohjana on Javascript ohjelmointikieli. Google Maps API on ei-kaupallisessa käytössä ilmainen. Sen saa käyttöönsä rekisteröitymällä Google-tilin käyttäjäksi, jonka jälkeen Google lähettää käyttäjälle domain-kohtaisen käyttöavaimen, joka sijoitetaan web-sivulle upotettavan koodin joukkoon omalle paikalleen. /11/

3 GSM/GPRS -MODUULI

Italialaisen Telitin valmistama GE863-GPS on markkinoiden pienin täyden 20-kanavaisen GPS-toiminnallisuuden sisältävä GSM/GPRS -moduuli. Se sisältää siis GSM-antennin lisäksi yksisiruisen GPS-vastaanottimen, joka pystyy tarkkailemaan 20:tä satelliittia samanaikaisesti paikannustarkkuudella 2,5 m. Sen fyysiset mitat ovat 41,4 x 31,4 x 3,6 mm. ja painoa laitteella on vain 9 grammaa. /12/

Moduulissa on TCP/IP -rajapinta, mikrokontrolleri sekä SIM-korttipaikka, johon käy minkä tahansa operaattorin kortti. Moduuli pystyy siten toimimaan ja ottamaan yhteyttä matkapuhelinverkossa, kuten tavallinen matkapuhelin. Käyttäjä tunnistetaan SIM-kortilla sijaitsevan yksilöllisen IMSI-avaimen avulla ja laitteen oma IP-numero saadaan aina sitä tarvittaessa operaattorilta /13, 14, s.144/. Kun laitteella vastaanotetaan tai lähetetään tietoa, varataan sitä varten fyysinen liikennöintikanava, mutta muutoin se vapautetaan muille GPRS-käyttäjille /14, s.170/. /12/

3.1 NMEA 0183

NMEA 0183 on National Marine Electronics Association -yhdistyksen kehittämä standardi erilaisten merielektroniikkalaitteiden keskinäistä viestintää varten, jota myös GE863-GPS -moduuli käyttää. NMEA 0183 käyttää yksinkertaista ASCII-merkeillä toteutettua viestintäprotokollaa, joka määrittelee, miten data siirretään lauseen sisällä lähettäjältä vastaanottajalle. Sovellustasolla standardi määrittelee myös viestin tyypin, jotta kaikki vastaanottajat pystyvät jäsentämään viestin oikein ja tarkasti. /15, 16/

3.1.1 NMEA 0183 -viestin rakenne

Kaikilla NMEA 0183 -viesteillä on samanlainen rakenne. Viesti alkaa $-merkillä ja seuraavat viisi merkkiä määrittelevät sen tyypin. Sen jälkeen seuraa data, jonka kaikki kentät on eroteltu pilkulla. Datan lopettaa *-merkki, jota seuraa vielä tarkistussumma, joka itse asiassa on XOR-operaation tulos kaikesta $- ja * merkkien välillä. /15, 16/

Esim. $GPGLL,4816.45,N,12311.12,W,225444,A,*1D

$	Aloitusmerkki
GP	Lähettäjä ID: GPS-unit
GLL	Viestityyppi: Geographic position, Latitude and Longitude
4816.45,N	Leveyspiiri: 48 astetta 16.45 minuuttia (N = Pohjoista leveyttä)
12311.12,W	Pituuspiiri: 123 astetta 11.12 minuuttia (W = Läntistä pituutta)
A	Data aktiivinen (tai mitätön V = Void)
*1D	Tarkistussumma (XOR-operaatio kaikesta välillä $ - *)

3.2 Kontrollointi

Laitetta voidaan kontrolloida sekä sisältä että ulkoa. Ulkoa tulevat komennot voidaan lähettää moduulille joko sarjaportin kautta tai langattomasti käyttäen hyväksi GSM/GPRS -ominaisuuksia. Moduuli tottelee NMEA-pohjaisille laitteille tyypillisiä AT-komentoja, joita käsitellään tarkemmin luvussa 3.2.1. Koska laite sisältää Python-suoritusympäristön, sen asetuksiin ja ominaisuuksiin päästään vaikuttamaan myös sisäpuolelta. Python-kielellä laitteen sisälle määritellyillä asetuksilla saadaan luotua hyvä perusta yhteyksien ja tietoliikenteen ohjaamiseen. /12/

3.2.1 AT-komennot

AT-komennot saivat alkunsa, kun yhdysvaltalainen modeemivalmistaja Hayes Microcomputer Products kehitti 1980-luvulla komentostandardin uutta Smartmodem-laitettaan varten. Vielä nykyäänkin lähes kaikkien analogisten modeemien komentoliikenne perustuu tuohon samaan standardiin tai sen muunnelmiin. /17, 18/

Standardin ideana oli yhdistellä muutaman merkin tekstipätkiä, joista muodostui yksi kokonainen komento /19/. Komentojen nimeenkin otettu lyhenne AT on yksi tällainen komennon osa. AT-komennoissa se tarkoittaa huomiota (attention) ja komentoviestin alussa tämän merkkijonon saadessaan modeemi tietää, että seuraavaksi on

tulossa komento. Siksi jokainen komentolause komentotilaan palaamista ja komennon toistoa lukuun ottamatta aloitetaan merkkijonolla AT (tai at). /20/

Esimerkiksi soittaminen modeemista toiselle onnistuu AT-komennolla ATDT1234567. Komennossa AT on siis aloitusmerkkijono, joka käskee modeemin valmistautua komentoon. D tarkoittaa puhelinnumeron valitsemista (Dial a telephone number) ja T äänivalintaa (Tone dial). Lopussa oleva numerosarja on sitten vastaanottavan modeemin puhelinnumero. Vastaanottava modeemi vastaa soittoon komennolla ATA (Answer incoming call). Yhteys on tämän jälkeen päällä. Lopettaakseen yhteyden on ensin palattava komentotilaan komennolla +++, jonka jälkeen yhteys saadaan poikki komennolla ATH (Hang-up). /20/

GE863-GPS:n komennot perustuvat alkuperäiseen Hayesin standardiin, mutta mukana on tietysti runsaasti Telitin omia komentoja, joilla nykyisenkin teknologian laitteita voidaan ohjata ja jotka sopivat juuri kyseisen laitteen ohjaukseen.

4 GPS

Viralliselta nimeltään NAVSTAR GPS (NAVigation Satellite Time And Ranging Global Positioning System) on maailmanlaajuinen, 24h vuorokaudessa toimiva, satelliitteihin perustuva paikannusjärjestelmä. Yhdysvaltain puolustusministeriö perusti sen alun perin sotilaskäyttöön, mutta järjestelmä on nykyisin käytössä myös siviileille. Järjestelmä on yksisuuntainen eli käyttäjä ei lähetä mitään signaalia satelliittiin päin. GPS-vastaanotin havaitsee satelliittien lähettämät signaalit ja laskee niiden avulla oman paikkansa. GPS-järjestelmän käytöstä vastaa Yhdysvaltain ilmavoimien avaruushallinto USAF Space System Division ja siviilikäytön yhteysvirastona toimii Yhdysvaltain liikenneministeriö /21, s.25/. /22, s.11/

4.1 GPS:n historia

Satelliittipaikannuksen historia alkaa jo 1960-luvulta, jolloin amerikkalaiset käynnistivät laivastonsa ja ydinsukellusveneidensä ohjaamiseen tarkoitetun Transit-järjestelmän. Itse asiassa navigointijärjestelmää ideoitiin jo 1940-luvulla, mutta tuolloin vielä avaruus- ja tietotekniikan kehittymättömyys olivat suunnitelmien esteenä. Transitiin kuului vain kuusi hiukan yli tuhannen kilometrin korkeudessa kiertävää satelliittia, joten se toimi vain lyhyissä jaksoissa ja osassa maapalloa. Hyvissä olosuhteissa sen tarkkuus oli 35–100 metriä. /21, s.21/

1970-luku toi mukanaan uusia satelliittipaikannusjärjestelmiä, kun Neuvostoliitto käynnisti oman armeijansa tarpeisiin suunnitellun Tsikadan 1976 ja ranskalais-amerikkalainen Argos aloitti toimintansa muutamaa vuotta myöhemmin. Argos tosin suunniteltiin palvelemaan paikannuksen lisäksi myös maailman laajuisen tieteellisen tiedon keräämisessä. Paikannustarkkuudeltaan Argos oli hyvin vaatimaton, jäädessään noin 500 metriin. /21, s.21/

Samalta vuosikymmeneltä alkaa myös itse GPS:n historia. Vuonna 1973 Yhdysvaltain puolustusministeriö teki päätöksen uuden järjestelmän tilaamisesta. Tavoitteena oli kehittää entistäkin tarkempi, tosiaikainen ja vihollisen häirinnän sietävä järjestelmä, joka palvelisi aseiden suuntaamista ja joukkojen ohjaamista. Vuonna 1978 avaruuteen laukaistiin ensimmäinen GPS-satelliitti Navstar 1. Vuoteen 1985 mennessä ensimmäisen sukupolven satelliitteja oli avaruudessa jo neljä. /21, s.23–25/

GPS:n seuraava vaihe alkoi, kun jo vuonna 1979 tilattujen toisen sukupolven Block II -satelliittien lähettäminen avaruuteen aloitettiin 1989. Vuoteen 1993 mennessä avaruuteen oli laukaistu viisi toisen sukupolven satelliittia ja samalla GPS-järjestelmä saavutti alustavan operatiivisen valmiuden. Avaruudessa oli siis 24 paikannuksessa toimivaa satelliittia, joista tosin osa oli vielä ensimmäisen sukupolven Block I -sarjaa. Huhtikuussa 1995 GPS-järjestelmä saavutti täyden operatiivisen valmiuden, kun käytössä oli 24 toisen sukupolven Block II -satelliittia. /21, s.25/

GPS-paikannustarkkuuden kannalta merkittävä muutos tapahtui vuonna 2000, kun siviilipaikantimien tahallinen häirintä SA (Selective Availability) kytkettiin pois

päältä presidentti Bill Clintonin määräyksestä. Sen seurauksena paikannustarkkuus parani suorastaan hämmästyttävälle tasolle aiemmasta parhaimmillaan n. 30-50 metrin tarkkuudesta muutamaan metriin. Tarkkuuden paranemisen ansiosta GPS-paikannusta pystytään nykyisin käyttämään tarkassakin paikannuksessa, kuten autojen navigaattoreissa. Periaatteessa Yhdysvaltain hallituksella on kuitenkin milloin tahansa mahdollisuus käynnistää häirintä uudelleen, jos se kokee siviilitarkkuudesta olevan sille uhkaa. /21, s.48–49/

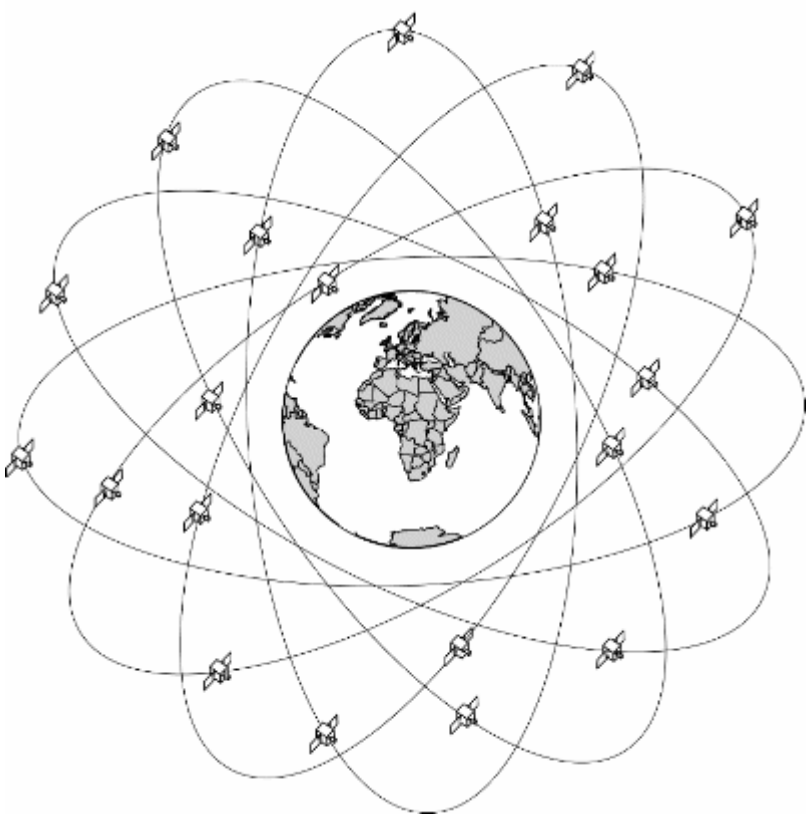
4.2 GPS-järjestelmä

Maailmanlaajuinen GPS-järjestelmä laitteineen voidaan jakaa kolmeen osaan ymmärryksen helpottamiseksi. Ensimmäinen ja tärkein osa, avaruusosa kostuu satelliiteista. Toinen on valvontaosa, joka koostuu maassa sijaitsevista keskusasemasta ja maa-asemista. Kolmatta osaa edustavat kaikki miljoonat GPS-paikantimet. /21, s.32–33/

4.2.1 Avaruusosa

GPS-järjestelmän avaruusosaan kuuluu 28 satelliittia, joista 24 on jatkuvassa operatiivisessa käytössä. Loput neljä satelliittia kiertävät maata siltä varalta että jokin operatiivisista satelliiteista tulee epäkuntoon. Siten operatiivisten satelliittien aukoton verkko pysyy jatkuvasti aukottomana. /21, s.33/

GPS-satelliitit kiertävät maapalloa yli 20 000 km korkeudessa kuudella eri ratatasolla, joilla jokaisella on neljä satelliittia. Kiertoratojen väli on 60 astetta ja niiden kaltevuuskulma päiväntasaajaan nähden on 55 astetta eli kiertoradat eivät ulotu leveyspiirien 55º etelä- tai pohjoispuolelle. Satelliittien täsmällinen kiertoaika on 11 tuntia 58 minuuttia, eli puoli tähtivuorokautta. Karkeasti ottaen satelliitit kiertävät siis maapallon kaksi kertaa vuorokaudessa. /21, s.33/



Kuva 1: GPS-satelliittien kiertoradat. /23/

Satelliitit ja niiden radat on sijoitettu siten, että joka puolella maapalloa olisi aina riittävän monta satelliittia käytettävissä. Lisäksi niiden kiertokorkeus on edullinen, koska matalalla kiertäessään niitä pitäisi olla huomattavasti enemmän kattaakseen koko maapallon. Samalla satelliittien etäisyys maapallosta on niin suuri, etteivät ylimmätkään ilmakehän kerrokset vaikuta niiden kiertoratoihin. Pieniä, maasta, kuusta sekä aurinkotuulesta johtuvia muutoksia kiertoratoihin tapahtuu kuitenkin jatkuvasti. Nykyisen sijoittelun ansiosta horisontin yläpuolella on aina vähintään kolme satelliittia, jotka GPS vaatii toimiakseen. Suomen taivaalla niitä on yleensä 7-8, mutta parhaimmillaan jopa 12 kappaletta. /21, s.34–35/

Nykyisistä satelliiteista pääosa on Navstar GPS / Block II -satelliitteja, jotka ovat satelliiteiksi hyvin pieniä, vain hieman yli 1 600 kg painavia ja 5,3 m^2 pinta-alaltaan ja ne kiertävät maata lähes 14 000 km/h nopeudella. Energiansa ne saavat kahdesta piiaurinkopaneelista, joiden yhteenlaskettu pinta-ala on noin 5 m^2. Lisäksi satelliitit sisältävät kaksi cesium- ja kaksi rubidium-kelloa, jotka tahdistetaan täsmälleen samaan aikaan muiden kanssa. Satelliittien suunniteltu toiminta-aika on 7,5 vuotta, jonka jälkeen ne korvataan tarvittaessa uudella. Satelliittien uusin sukupolvi syrjäyttää vanho-

ja Block II -satelliitteja pikku hiljaa, ja samalla niiden toiminta-aika lähes tuplaantuu. Myös lukumäärä nousee lopulta 33:een entisen 28:n sijaan vuoden 2012 loppuun mennessä nostaen siten GPS-järjestelmän luotettavuutta entisestään. /21, s.35–36/

4.2.2 Valvontaosa

GPS-järjestelmän valvontaosa koostuu keskusasemasta, viidestä maa-asemasta ja neljästä maa-antennista. Kaikki osat sijaitsevat maassa ja ovat välttämättömiä järjestelmän jatkuvan valvonnan ja huoltamisen takia. /21, s.39/

Koko GPS-järjestelmän valvonta ja ohjaus on keskitetty keskusasemalle, joka sijaitsee Schrieverin lentotukikohdassa Yhdysvaltain Coloradossa. Järjestelmän käytännön operoinnista vastaa Yhdysvaltain ilmavoimien 2. avaruuslennoston 50. avaruuslaivue, joka tarkkailee jatkuvasti satelliittien kiertoratoja ja kelloja sekä niissä tapahtuvia muutoksia. Korjauspäivitykset tehdään sitten laskemalla poikkeamat alkuperäisistä tiedoista. Jatkuva päivittäminen on elintärkeää järjestelmän tarkkuuden ylläpitämiseksi, sillä jo sadasosasekunnin ero kelloissa voi aiheuttaa paikanmääritykseen virhettä jopa 2800 km /21, s.47/. /21, s.39–40/

Maa-asemia on viisi ja ne on sijoitettu lähelle päiväntasaajaa eri puolelle maapalloa. Ne sijaitsevat Havaijilla, Kwajaleinilla Tyynellä valtamerellä, Diego Carcialla Intian valtamerellä, Ascensionilla Atlantilla sekä Colorado Springsissä. Ne keräävät tietoja avaruuden tapahtumista ja ovat jatkuvasti suorassa yhteydessä keskusasemaan. Myös keskusasemalla korjattuja päivityksiä kulkee jatkuvana virtana maa-asemien kautta maailman jokaiseen yksittäiseen GPS-vastaanottimeen. /21, s.40–41/

Maa-asemien omat GPS-vastaanottimet mittaavat jatkuvasti etäisyyttä taivaalla näkyviin satelliitteihin ja vertaavat omien atomikellojensa käyntiä satelliittien vastaaviin. Atomikelloja verrataan samalla koko järjestelmän pääkelloon, Yhdysvaltain laivaston kelloon Floridassa. GPS-ajaksi (GPST) kutsuttu aika eroaa muutamia sekunteja aurinkoon sidotusta keskiaurinkoajasta, joka tunnetaan nimellä GMT (Greenwich Mean Time) /21, s.47/. /21, s.40–41/

Neljän maa-antennin tehtävänä puolestaan on välittää maa-asemien keräämää tietoa luotettavasti kaikille avaruusosan satelliiteille. Antennien kautta lähetetään myös komentoja satelliiteille ja niiden kautta otetaan vastaan satelliittien tilatietoja. Ne on sijoitettu ympäri maapalloa koko maapallon kattamiseksi. /24/

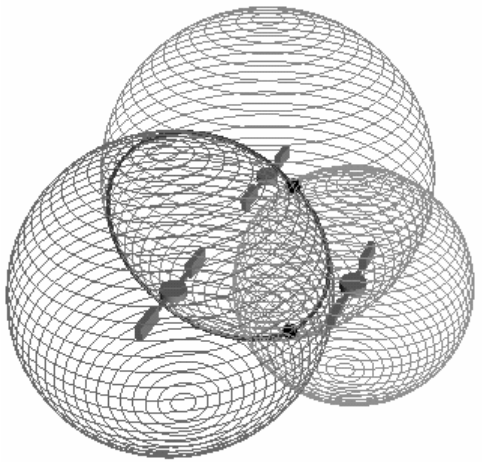
4.2.3 Paikantimet

Kolmannen osan GPS-järjestelmästä muodostavat kaikki miljoonat GPS-vastaanottimet maailmassa. Vastaanotin on se osa, joka tekee varsinaisen paikanmäärityksen. Se mittaa etäisyyden satelliitteihin kertomalla signaalin matkaan kuluneen ajan valonnopeudella ja laskee sitten oman sijaintinsa usean satelliitin sijainnin perusteella. Erilaisia sovelluksia vastaanottimista on kymmeniä ja useisiin eri tarpeisiin. Nykyisin tutuimpien autonavigaattorien lisäksi omat sovelluksensa on niin käsikäyttöön kuin meriliikenteellekin. /21, s.33,s.45/

4.3 Paikannustekniikka

GPS-paikannuksen toiminta perustuu satelliittien ja paikantimen välisen etäisyyden avulla tehtävään kolmiomittaukseen eli trilateraatioon. Satelliittien signaaleista muodostuu avaruuteen valtavat pallot, joiden keskipisteinä ovat itse satelliitit. Kun kahden satelliitin signaalipallot menevät jollakin hetkellä osin sisäkkäin, niiden pintojen leikkauskohta muodostaa suuren ympyrän. Sillä hetkellä paikantimen voidaan sanoa olevan missä tahansa kohdassa ympyrän kehällä. /21, s.42–43/

Kun otetaan mukaan vielä kolmannenkin satelliitin signaalista syntyvä pallo, ja se leikkaa kahden aikaisemman pallon leikkauskohtaan syntyneen ympyrän, saadaan kaksi mahdollista pistettä, jossa paikannin voi olla. Toinen pisteistä voi olla kaukana avaruudessa tai hyvin syvällä maapallon sisällä ja toinen jossakin kohtaa maapallon pinnalla. Vaikka oikea piste olisi helppo myös päätellä, GPS-järjestelmässä valinta perustuu hyvin tarkkoihin laskuihin /21, s.42/. /21, s.44/



Kuva 2: Signaaleista syntyvien pallojen leikkauspisteet. /25/

GPS-paikannin tulee toimeen jollakin tavalla kolmen satelliitin signaaleilla, mutta toimii sitä paremmin, mitä useamman satelliitin avulla se voi tehdä laskujaan. Osatakseen laskea korkeutta, paikannin tarvitsee vielä signaalin neljännestä satelliitista, josta se saa mm. kellonkorjaustietoa. GPS:n tarkkuus on kuitenkin sitä parempi, mitä suurempi osa taivaasta jää signaalia antavien satelliittien keskelle. /21, s.44/

Tarkan sijainnin selvittämiseksi on vielä mitattava etäisyydet satelliiteista paikantimeen maapallon pinnalle. Etäisyys saadaan laskettua hyvin yksinkertaisella kaavalla $s = v \cdot t$. Kun tiedetään signaalin nopeus v ja matka-aika t, saadaan laskettua etäisyys s. Signaali kulkee valonnopeudella, joten sen nopeus on 299 792,5 km/h. Satelliitti ja paikannin luovat molemmat samalla hetkellä ja täsmälleen samanlaisen näennäissatunnaisen PRN-koodin. Kun lasketaan koodien vastaavien kohtien ajallinen ero, saadaan viive, joka on samalla signaalin kulkemiseen käyttämä aika. Kertomalla nämä suureet, saadaan laskettua paikantimen ja satelliitin välinen etäisyys. /21, s.44–45/

4.4 Signaali

GPS-satelliiteista lähteviä signaaleja on kahta tyyppiä, siviili- ja sotilaskäyttöön omansa. Signaalit on erotettu kahdeksi eri signaaliksi, jotta sotilaallisen kriisin aikana Yhdysvaltain vastustajat eivät hyötyisi Yhdysvaltain omasta järjestelmästä, mutta oma hyöty säilyisi edelleen. /21, s.39/

Sotilasosaa, PPS:ää (Precise Positioning System) saavat hyödyntää USA:n armeijan lisäksi vain Yhdysvaltain hallituksen valtuuttamat käyttäjät, muut amerikkalaiset viranomaiset ja Yhdysvaltain liittolaiset. Heille tarkoitettu alemman tason signaali P-code on salattu, ja sen purkamiseen tarvitaan GPS-paikantimen sotilaallinen versio. Sotilassignaalin paikannustarkkuus on vakaasti pari metriä. /21, s.39/

Ylemmän taajuuden signaali SPS (Standard Positioning Service) on GPS:n siviiliosa. Se on kaikkien vapaassa, ilmaisessa käytössä. Järjestelmän ylläpitäjä on kuitenkin tarkoituksella jättänyt itselleen mahdollisuuden häiritä siviilipaikannusta monellakin tavalla. Tärkein näistä on tahallinen paikannustarkkuuden häirintä SA (Selective Availability), joka on järjestelmän sisälle rakennettu tekninen menetelmä, jolla pystytään vaikuttamaan kaikkiin C/A-koodia lukevien paikantimien tarkkuuteen manipuloimalla atomikelloja /21, s.48/. /21, s.39/

Itse signaali, koostuu useasta osasta. Kantoaaltoon on moduloitu sekä tietosisältö että koodi. Niin sanotun C/A-koodin (Coarse Acquisition Code) ansiosta GPS paikannin voi erottaa kaikki satelliitit toisistaan siitä huolimatta, että ne toimivat samalla taajuudella. /21, s.37–38/

Tieto-osa on 1500 bittiä pitkä ja se lähetetään joka 30. sekunti 50 bitin sekuntinopeudella. Siihen on pakattu binäärijonoiksi mm. satelliittien ratatiedot ja kunnon sisältävä almanakka, ilmakehän ylimmän kerroksen muutokset sekä maa-asemien päivittämät tiedot satelliittien sijainnista. Tärkeimpänä sisältönä ja koko paikantamisen perustana on kuitenkin äärimmäisen tarkka tieto kellonajasta, jolloin signaali lähti satelliitista. /21, s.38/

Tunnistusta varten signaali on muokattu muistuttamaan satunnaista radioaaltojen taustakohinaa. Todellisuudessa se on kuitenkin tuotettu erittäin tarkoilla matemaattisilla menetelmillä. Siksi signaalia kutsutaan näennäissatunnaiseksi PRN-koodiksi (Pseudo Random Noise Code). /21, s.38/

4.5 Maantieteelliset koordinaatit

Sijainti ilmoitetaan maantieteellisten koordinaattien, leveyden ja pituuden avulla. Maantieteellinen leveys, josta käytetään kansainvälisesti nimeä latitudi, ilmaisee paikan maapallon pinnalla pohjois-eteläsuunnassa. Leveyspiirit ovat siis maapallon ympärille kuviteltuja ympyröitä, jotka ovat samansuuntaisia päiväntasaajan kanssa. Jos maapallo ajatellaan ympyräksi, jonka täysi kehä on 360º, paikan sijainti ilmaistaan ympyrän kehän osina eli asteina, asteen osina eli minuutteina ja minuutin osina eli sekunteina. Leveyspiireissä päiväntasaaja vastaa 0º, pohjoisnapa 90º pohjoista leveyttä ja etelänapa 90º eteläistä leveyttä. /21, s.152–153/

Maantieteellinen pituus, josta käytetään usein kansainvälistä nimeä longitudi, ilmaisee paikan maapallon pinnalla itä-länsisuunnassa. Pituuspiirit ovat siis puolestaan napojen kautta kulkevia kuviteltuja ympyröitä maapallon ympärillä. Koska ne kulkevat sekä päiväntasaajan että napojen kautta, on meridiaanien välimatka suurempi lähellä päiväntasaajaa kuin napojen lähettyvillä. Pituuspiirien välimatkat eivät siis ole metreissä yhtä suuria joka puolella, niin kuin leveyspiirien välimatkat ovat. Pituuspiiri 0º eli nollameridiaani kulkee Lontoossa sijaitsevan Greenwichin tähtitornin kautta ja jakaa maapallon itäiseen ja läntiseen puoliskoon, joista molemmat ovat 180º. /21, s.154–155/

5 ETÄHALLINTASOVELLUS

Sovelluksen ideana on mahdollistaa valmiiksi rakennetun GSM/GPRS -moduulin etähallinta ja konfigurointi PHP-pohjaisena web-sovelluksena. Samalla sovelluksen tulisi olla kuitenkin helppokäyttöinen ja turvallinen käyttää, jotteivät asiattomat käyttäjät pääse laitteeseen käsiksi etäältä. Sovelluksen ohjelmointikielenä on käytetty XHTML:ää ja PHP:tä lukuun ottamatta karttasovelluksessa olevaa dynaamista Google Maps -karttaa, joka on toteutettu Javascriptillä.

5.1 Autentikointi

Kaikkiin sovelluksen PHP-sivuihin pääsyyn vaaditaan sisään kirjautuminen. Jos käyttäjä ei ole kirjautuneena sisälle eli sille ei ole avointa istuntoa käynnissä, selain ohjautuu jokaiselta sivulta kirjautumissivulle, joka tässä tapauksessa on itse /index.php.

```
if(!session_is_registered('member_ID')) {
  header('Location: index.php?msg=requires_login');
```

Kirjautumissivu on toteutettu yksinkertaisena lomakkeena, joka lähettää käyttäjän syöttämät arvot post-metodilla erilliselle tunnusten tarkistussivulle.

```
<?php
session_start();
include('mysql_yhteys.php');
if(isset($_POST['submit'])){
   $username = strip_tags($_POST['username']);
   $password = md5(strip_tags($_POST['password']));
   $query = sprintf("SELECT ID FROM members WHERE username =
'%s' AND user_password = '%s' LIMIT 1;",
addslashes($username), addslashes($password));
   $result = mysql_query($query);
   if(1 != mysql_num_rows($result)){
       header('Location: index.php?msg=login_failed');
      }
   else{
       $row = mysql_fetch_assoc($result);
       $_SESSION['member_ID'] = $row['ID'];
       header('Location: admin.php');
      }
}
?>
```

Aloitetaan istunto ja avataan yhteys tietokantaan. Tietokantayhteyden avaus on ulkoistettu toiselle sivulle, jotta se voidaan helposti sisällyttää kullekin sivulle aina sitä tarvittaessa. Mikäli kirjautumissivulla kysytyt tiedot on syötetty, verrataan käyttäjän syöttämää tunnusta ja syötetyn salasanan MD5-tiivistettä tietokannan käyttäjätaulusta

löytyviin. Myös tietokannassa olevat salasanat ovat siis MD5-tiivistettyjä. Jos käyttäjätaulusta löytyy vastaavat sisällöt, käyttäjän selain ohjautuu eteenpäin valikkosivulle nimeltä admin.php ja käyttäjätiedot tallennetaan myöhempää käyttöä varten. Muutoin kirjautuminen epäonnistuu, selain palaa takaisin kirjautumissivulle ja epäonnistuminen osoitetaan myös osoiterivillä.

Autentikointi moduulin puolella on hoidettu siten, että laite ei ota vastaan yhteyksiä muilta kuin admin-palvelimelta, jossa sovelluskin sijaitsee. Tämä on määritelty Python-kielellä laitteen sisälle tehdyssä perusohjelmassa, joka toimii samalla kuten palomuuri.

```
res = MDM.receive(1)
res = MDM.send('AT#FRWL=1,"xxx.xxx.xxx.xxx",
"xxx.xxx.xxx.xxx"\r',0)
res = MDM.receive(100)
```

Komennolla ”AT#FRWL” määritellään yhteyksiin sallitun palvelimen ip-osoite ja aliverkon peite. Yhteydet muista osoitteista estetään. Tämän avulla varmistetaan, etteivät asiattomat pääse ohjaamaan laitetta.

5.2 MySQL-tietokanta

MySQL oli ehdoton valinta tietokannaksi sovellukselle sen PHP-soveltuvuuden takia. PHP:ssä on valmiit funktiot MySQL:n käsittelyyn ja kyselyjen tekemiseen, mikä tekee niistä yhdessä hyvän parin tietokannan ja sovelluksen väliseen toimintaan. MySQL oli lisäksi admin-palvelimella valmiiksi asennettuna, vaikka asennettu versio 3.23.49 onkin julkaistu jo vuonna 2002 /26/.

Aina, kun sovellus tarvitsee yhteyden tietokantaan, se avataan erikseen siihen tarkoitetun mysql_connect -funktion avulla. MySQL-yhteyden ottaminen on yleensä, ja tässäkin työssä erotettu omaksi PHP-tiedostokseen, joka sitten vain sisällytetään include()-funktion avulla joka sivulle, missä yhteyttä tarvitaan. Yhteydenmuodostus ja tietokannan valinta tapahtuu PHP:ssä seuraavasti (muuttujilla kuvitteelliset arvot):

```
<?php
$host = "mysql.foohost.com";
$dbname = "database_name";
$dbpass = "password";

$connection = mysql_connect("$host","$dbname","$dbpass");
$db = mysql_select_db("$dbname",$connection);
?>
```

Kun MySQL-yhteyttä ei enää tarvita, yhteys voidaan sulkea mysql_close()-funktiolla. Yksinkertainen kysely tietokannasta tehdään mysql_query()-funktion avulla ja tulos tallennetaan muuttujaan:

```
$result = mysql_query("SELECT * FROM table", $connection);
```

Jo avattu yhteys tietokantaan sisällytetään kyselyyn muuttujaan tallennettuna parametrina. Mikäli yhteys katkaistaan välillä koodissa mysql_close()-funktiolla, se pitää avata uudelleen, ennen kuin kysely voidaan suorittaa.

5.2.1 phpMyAdmin

Tietokannan taulujen muokkaus toteutettiin phpMyAdmin-hallintatyökalun avulla, jonka versio 2.6.0 oli asennettuna palvelimelle. Siinä on graafinen käyttöliittymä ja työkalut taulujen tekemiseen ja muokkaamiseen. Siksi erillistä sivua tietokantojen muokkaamiseen ei tarvita.

5.2.2 Tietokantataulut

Tietokannassa on etähallintasovellusta varten kaksi taulua, joista toinen sisältää tiedot kirjautumista varten ja toinen varsinaiseen työhön liittyviä tietoja. Tauluissa on käytetty tietokantamoottorina InnoDB:tä oletusarvoisen MyISAM:in tilalla. MyISAM-moottorissa ei ole tukea transaktioille, jotka mahdollistavat paremman palautumiskyvyn virheistä ja eristävät ohjelmat tietokannasta. InnoDB:stä puolestaan löytyy transaktiotuki. Lisäksi InnoDB käyttää parempaa rivitason lukitusta taulun

muokkaamisen aikana, kun MyISAM lukitsee koko taulun. Rivitason lukitus on eduksi, kun taulun monia taulun rivejä päivitetään usein. /27/

Kirjautumista varten luotiin taulu nimeltä ”members”. Taulu sisältää kolme saraketta, jotka ovat ID, username ja user_password. Seuraavasta taulukosta käy ilmi sarakkeiden tietotyypit ja muut tiedot:

Taulukko 1: Sarakkeet ”members” -taulussa.

Sarake	**Tyyppi**	**Lisätiedot**
ID	int(max 5 merkkiä)	Primary_key, Not_null, Auto_increment
username	varchar(30)	Not_null
user_password	varchar(30)	Not_null

ID lisääntyy siis automaattisesti yhdellä jokaisella rivillä ja toimii ensisijaisena avaimena. Salasanojen arvot lisätään tauluun MD5-tiivistetyssä muodossa. Yksikään taulukon arvoista ei voi olla tyhjä eli ”null”.

Toinen taulu, nimeltään ”laitteet”, sisältää kaiken muun sovelluksessa tarvittavan tiedon. Taulussa on siis jokaiselle tulevalle laitteelle oma rivinsä. Seuraavassa taulukossa on kuvattu ”laitteet” -taulukon sarakkeet, niiden tietotyypit ja lisätiedot:

Taulukko 2: Sarakkeet ”laitteet” -taulussa.

Sarake	**Tyyppi**	**Lisätiedot**
item_id	int(10)	Primary_key, Not_null
item	varchar(20)	Not_null
info	varchar(100)	
last_reg_time	datetime	
ip	varchar(15)	

Item_id on siis itse määritelty yksikäsitteinen avain ja item on laitteen nimi. Info-kentässä on yleistä tietoa laitteesta ja sen sijainnista, ip sisältää laitteen ip-osoitteen ja last_reg_time kertoo koska laite on viimeeksi päivittänyt osoitteensa.

5.3 Moduulin konfigurointi

Laitteen sisällä on Python-kielellä kirjoitettu perusohjelma, joka mahdollistaa erilaisten asetusten määrittämisen laitteeseen. Jotta laitteen etäkomentaminen olisi mahdollista, se pitää määritellä pitämään auki yhtä porttia ja kuuntelemaan mahdollisia yhteydenottoja. Komennolla “AT#SL” (Socket Listen) määritellään laite kuuntelemaan porttia 1124. Laitteen kuudesta mahdollisesta porttiyhteydestä tämä asetetaan olemaan numero kolme.

```
res = MDM.receive(1)
res = MDM.send('AT#SL=3,1,1124\r',0)
res = MDM.receive(100)
```

Dynaamisen ip-osoitteen takia laitteen on myös ilmoitettava oma osoitteensa tietokantaan aina verkkoon tullessaan. Python-ohjelmassa on siksi myös määritelty, että laite suorittaa tällöin palvelimelta tiedoston storedns.php.

```
DNS_SERVER_STR="GET http://www.osoite.com/storedns.php"
```

PHP-sivu poimii suorittavan laitteen ip-osoitteen ja päivittää sen jälkeen tietokannan ”laitteet” -tauluun laitteen omalle riville sen uuden ip-osoitteen sekä viimeisimmän rekisteröitymisen päivämäärän ja ajan.

```
$oma_ip= $_SERVER["REMOTE_ADDR"];
$a = $HTTP_GET_VARS["a"];

$query = 'UPDATE `laitteet` SET `ip` = "' . $oma_ip . '" WHERE
`item` = "' . $a . '"';
echo "OK=",$result = mysql_query($query, $laitteet) or
die(mysql_error());
echo "\n";

$query = 'UPDATE `laitteet` SET `last_reg_time` = "' . date(
'Y-m-d H:i:s') . '" WHERE `item` = "' . $a . '"';
echo "OK=",$result = mysql_query($query, $laitteet) or
die(mysql_error());
echo "\n";
```

Kun laitteen porttiin 1124 otetaan yhteyttä, se hyväksyy yhteyden komennolla AT#SA, jonka jälkeen portti voi ottaa vastaan lähetettyä dataa. Tämän jälkeen laite lähettää toiselle osapuolelle tervehdysviestin ”HI!”. Mikäli laitteelle ei lähetetä mitään AT-komentoa, on tuloksena vain pelkkä tervehdys, jonka jälkeen yhteys lopetetaan.

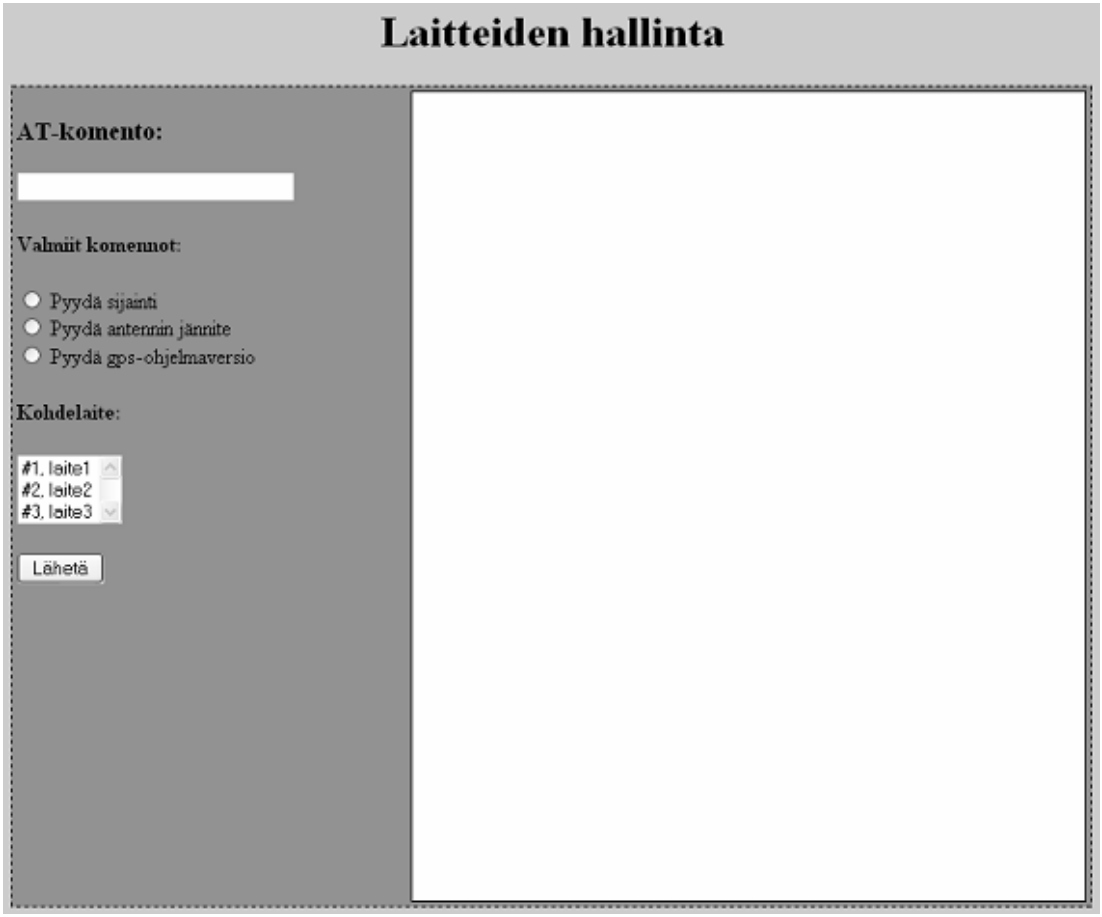
```
koputus = res.find('SRING: 3')

if koputus != -1 :
        res = MDM.send('AT#SA=3\r',0)
        res = MDM.receive(100)
        res = MDM.send('HI!\n',0)
```

Laitteen konfigurointiin on äärettömästi mahdollisuuksia. Asetusten määrittäminen perustuu kuitenkin sisäpuoleltakin AT-komentoihin, joten kaikki, mitä AT-komennoilla on määriteltävissä, voidaan toteuttaa myös sisällä olevalla Python-koodilla.

5.4 Hallintasivu

Kuten koko sovellus, laitteiden hallintasivukin on toteutettu PHP-ohjelmoinnilla. Se on nimeltään commandcentre.php. Sivun ulkoasu on yksinkertainen, sillä sivun tekemisessä haluttiin keskittyä toimivuuteen ja selkeyteen. Sivu voidaan jakaa näennäisesti kolmeen osaan, jotka ovat komento-osa, tulososa ja tausta. Vasemmassa reunassa olevassa komento-osassa on lomake, jonka tiedoilla haluttu laite antaa vastauksen oikean reunan tulososaan. Taustassa ei ole toiminnallisuutta.



Kuva 3: Laitteiden hallintasivu.

Ensin valitaan haluttu komento, jonka pitää olla laitteen ymmärtämä AT-komento. Sivussa on esiasetettuna kolme komentovaihtoehtoa, joita on mahdollista lisätä sen mukaan, mitä eniten tarvitaan. Tässä valmiina vaihtoehtoina ovat sijainnin, antennin jännitteen ja gps-ohjelmaversion kysyminen.

```
<form method="get" action="commandcentre.php">
<h3>AT-komento:</h3>
<input name="b1" type="text" size="30" /><br />
<h4>Valmiit komennot:</h4>
<input name="b2" type="radio" value="AT$GPSACP?" /> Pyydä sijainti<br />
<input name="b2" type="radio" value="AT$GPSAV?" /> Pyydä antennin jännite<br />
<input name="b2" type="radio" value="AT$GPSSW?" /> Pyydä gps-ohjelmaversio<br />
```

Mikäli haluttu AT-komento ei löydy valmiiden vaihtoehtojen joukosta, se voidaan antaa suoraan ylälaidan tekstilaatikkoon. Jos tekstimuotoinen komento on annettu, se tulkitaan lähetettäväksi komennoksi, eli mikäli haluaa valita valmiin vaihtoehdon, on tekstilaatikon oltava tyhjä.

```
if (empty($HTTP_GET_VARS["b1"]))    {
$komento = $HTTP_GET_VARS["b2"];
}
else {
$komento = $HTTP_GET_VARS["b1"];
}
```

Komennon lisäksi valitaan kohdelaite. Kaikki laitteet löytyvät tietokannan taulusta ”laitteet”, josta ne saadaan PHP:n avulla listattua valintalaatikkoon. Ennen tietokantakyselyä on kuitenkin taas avattava MySQL-yhteys. Laatikon jokaiselle riville tulostetaan laitteen id-numero ja nimi, jotta ne on helppo erottaa toisistaan valittaessa.

```
echo" <select name=\"a\" size=\"3\">  ";

$results = mysql_query("SELECT item_id, item from laitteet
ORDER BY item",$connection);
$id = "item_id";
$idname = "item";
echo mysql_error();

if (mysql_Numrows($results)>0)    {
  $numrows=mysql_NumRows($results);
  $x=0;
  while ($x<$numrows){
    $theId=mysql_result($results,$x,$id);
    $theName=mysql_result($results,$x,$idname);
echo "<option value=\"$theId\">#$theId, $theName</option>\n";
    $x++;
  }
}
echo "</select>";
```

Valintalaatikon aloittamisen jälkeen tehdään mysql-kysely. Kun on tarkistettu, löytyykö riveiltä sisältöä, rivit lasketaan. Sitten kaikkien löytyneiden rivien tiedot tallennetaan muuttujiin while-silmukan avulla. Lopuksi jokainen näistä tiedoista asetetaan <option>-tagien väliin, jonka jälkeen vielä lopetetaan valintalaatikko.

Kun attribuuteille on nyt saatu arvot, voidaan lomake lähettää. Lomakkeen tietojen siirrossa käytetään GET-metodia, joten arvot tulevat näkyviin myös osoiteriville ja palvelua voidaan käyttää myös syöttämällä attribuuttien arvot suoraan osoiteriville esimerkiksi muodossa /commandcentre.php?b2=AT$GPSACP?&a=3, jolla saataisiin vastaukseksi laitteen 3 sijainti.

Lomakkeen tiedoilla haettu tulos saadaan vastauskenttään oikeaan reunaan. Samalla jokainen tulos tai virhesanoma talletetaan lokitiedostoon nimeltä ”response.txt”. Tiedosto pitää ensin avata käyttäen fopen()-funktiota.

```
$tiedosto = fopen("response.txt","a+");
```

Samalla annetaan tiedoston käyttötapa, joka tässä on “a+”. Tässä moodissa tiedosto avataan lukemista sekä kirjoittamista varten ja osoitin asetetaan tiedoston loppuun. Siten uusin rivi on aina tiedoston viimeisellä rivillä. Myös nykyinen päivämäärä ja kellonaika tallennetaan muuttujaan myöhempää käyttöä varten.

```
$date = date("H:i:s d.m.Y");
```

Jotta laitteeseen saataisiin yhteys, on haettava sen ip-osoite tietokannan taulusta “laitteet”, johon laite on päivittänyt numeronsa käynnistyessään. Suoritetaan MySQL-kysely, jonka tuloksena saadaan halutun laitteen ip-osoite.

```
$query = 'SELECT ip FROM `laitteet` WHERE item_id = "' . $a .
'"';

if (isset($a))    {
echo $query . "\r\n";
echo "OK=",$result = mysql_query($query, $connection) or
die(mysql_error());
echo "\r\n";
```

```
$row = mysql_fetch_assoc($result);
$laiteip=$row['ip'];
echo $laiteip . "\r\n";
```

Tallennetaan ensin muuttujaan valmis kysely, jossa laitteen id:n arvona toimii lomakkeesta saatu arvo ”a”. Mikäli ”a” oli annettu, kysely suoritetaan ja se tulostetaan sivun vastausosaan vastauksesta poimitun ip-osoitteen kanssa. Kun laitteen ip on saatu selville, avataan yhteys laitteeseen fsockopen()-funktion avulla.

```
$fp = fsockopen($laiteip, 1124, &$errno, &$errstr, 30);
```

Yhteys avataan tietokantataulusta saatuun ip-osoitteeseen, porttiin 1124. Laitteeseen on siis jo valmiiksi määritelty, että se sallii tältä palvelimelta tulevat yhteydenottopyynnöt. Jos yhteyden muodostus ei onnistu 30 sekunnissa, se lopetetaan.

```
if (!$fp) {
            $tulos = "Error: Could not open socket
            connection.";
            echo "$errstr ($errno)<br />\n";
            fputs($tiedosto, "$date \r\n");
            fputs($tiedosto, "$errstr ($errno)\r\n");
            fclose($tiedosto);
            }

echo $tulos;
```

Epäonnistumisen jälkeen sivun vastausosaan tulostetaan syyn kuvaus ja virhekoodi sekä teksti ”Error: Could not open socket connection.”. Lokitiedostoon kirjoitetaan vastaavasti päivämäärä ja aika sekä virhetiedot, jonka jälkeen lokitiedosto suljetaan.

Kun yhteys laitteen porttiin saadaan muodostettua, päästään lähettämään käskyjä myös laitteelle. Samalla tulokseen saadaan näkyviin, mitä sanomien vaihdossa tapahtuu.

```
else {
      $tulos .=  "Device found!\r";
      fputs($tiedosto, "$date \r\n");
```

```
fputs($tiedosto, "Device found!\r\n");
sleep(1);
```

Ensin muuttujaan $tulos lisätään teksti ”Device found!” ja sama tehdään myös lokitiedostoon, mutta kirjoitetaan myös aika. Sovellusta testatessa huomattiin, että laite ei aina ehdi vastaamaan, kun useita pyyntöjä lähetetään peräkkäin samassa PHP-koodissa, vaan jumiutuu paikoilleen. Jotta tämä saataisiin ehkäistyä, lopetetaan välillä koodin suorittaminen sekunniksi funktiolla sleep().

```
$tulos.= fgets ($fp, 128);
$tulos.= " 1 ";
sleep(1);
fputs($fp, $komento . "\r\n");
```

Laitteen vastaus kysytään funktiolla fgets() ja lisätään muuttujaan $tulos. Vastauksen lukeminen päättyy, kun 127 (128-1) tavua on luettu. Lisätään muuttujaan vielä teksti ” 1 ” lähinnä vain tunnisteeksi sille, että laitteen yhteysvahvistus on saatu ja seuraavaksi vuorossa on vastaus annettuun komentoon. Kun ensin on taas odotettu sekunti, lähetetään aiemmin lomakkeella annettu komento laitteelle.

While-silmukalla varmistetaan vielä, että laitteelta saadaan kokonaan pitkäkin vastaus annettuun komentoon. Laitteelta tullut, onnistunut vastaus päättyy viestiin ”OK”, ja kun se sisältyy vastaukseen, on se kokonaan tullut. Vastauksen perään lisätään, kuinka monta kierrosta silmukan piti tätä varten käydä.

```
$i=0;
while ($i<10) {
      $temp = fgets ($fp, 1024);
      $tulos .= $temp;
      fputs($tiedosto,$temp);
      $i++;

      if (strstr($tulos, "OK")) {

            $tulos.=" /";
            $tulos.=$i;
            $tulos.="/ ";
            $i=12;
```

```
                }
            }
            fclose($tiedosto);
            fputs($fp, '---');
            fclose ($fp);
    }
    echo $tulos;
```

Lopuksi suljetaan vielä lokitiedosto ja yhteys moduuliin. Yhteys laitteeseen suljetaan sulkemalla sekä PHP:n fsockopen()-funktio että lähettämällä laitteelle lopetus komento ”---”. Tämän jälkeen tulostetaan saatu ja yhteen kerätty vastausjoukko näkymään sivun vastausosaan.

Kuva 4: Laitteelta saatu vastaus.

5.5 GPS-karttasovellus

GPS-karttasovellus tehtiin täydentämään laitteiden hallintasivua ja sen sijaintikyselyä. Tavoitteena oli samalla tuoda esille laitteen ja etähallinnan soveltamismahdollisuuksia. Sovellus käyttää hyväksi laitteelta saatuja koordinaatteja ja näyttää laitteen sijainnin kartalla. Karttana käytettiin sivuille upotettavaa, valmista ohjelmointirajapintaa, Google Maps API:a.

Sivun rakenne on hyvin samanlainen kuin itse hallintasivussa, mutta tässä sovelluksessa ei tarvita laitteelle lähetettävän komennon valintaa. Lisäksi laitteen vastauksen esittämiseen varattu tuloskenttä on korvattu tässä kartalla. Sen sijaan tietokantaan perustuva laitevalintaluettelo on samanlainen, kuin laitehallintasivussa.

Kuva 5: Karttasovellus.

Ilman annettuja koordinaatteja kartta ja sijainnin osoitinmerkki ovat keskitettynä kartan keskipisteeseen. Kartassa on myös mahdollisuus liikkua sekä tasossa että syvyyssuunnassa joko vasemmassa yläkulmassa olevien navigointinäppäimien avulla tai hiirellä tarttuen ja vetäen.

5.5.1 Kartan upottaminen sivulle

Google Maps API:n saa upotettua sivulle helposti. Kun on saanut käyttöönsä avainmerkkijonon Googlen vaatiman rekisteröitymisen seurauksena, on kartta valmis käy-

tettäväksi. Kartan osalta koodi on javascript-ohjelmointikieltä, ja sen määrittelyt sijoitetaan PHP-sivun <head>-osioon. Alkuun määritellään kieli ja annetaan samalla Googlelta saatu avainmerkkijono koodin lähteeksi.

```
<script src="http://maps.google.com/maps?file=api&amp;v=2&amp;
key=Xx1xX&sensor=true" type="text/javascript"></script>
```

Tässä koodissa merkein “Xx1xX” merkitty avain on siis todellisuudessa paljon pidempi, Googlen avaingeneraattorin luoma merkkijono, joka sisältää sekä numeroita että isoja ja pieniä kirjaimia. Nykyisin Google myös edellyttää, että koodissa kerrotaan, mikäli GPS-paikanninta käytetään käyttäjän sijainnin määrittämiseen /28/. Tämän takia avaimen perässä on siis määritelmä ”sensor=true”.

Laitteen valinta ja lomakkeen tietojen lähettäminen tapahtuu siis karttasovelluksessa muuten täsmälleen samalla tavalla kuin hallintasivussakin, mutta tässä lähetettävä komento ei ole valinnainen, vaan valmiiksi määritelty. Laitteen sijainnin kysyvä AT-komento on ”AT$GPSACP?”, mutta jotta $-merkkiä ei tulkittaisi muuttujan aloitusmerkiksi, on komentoon lisättävä kenoviiva ennen $-merkkiä.

```
$komento = "AT\$GPSACP?";
```

5.5.2 Koordinaattien muokkaus

Laitteen antamassa vastauksessa on karttasovelluksen kannalta paljon turhaa sisältöä, koska ainoat tarvittavat arvot ovat pituus- ja leveyspiirikoordinaatit. Vastauksessa olevat eri tiedot on kaikki eroteltu pilkulla toisistaan, joten ne pystytään helposti erottamaan toisistaan. Vastaus sisältää seuraavassa järjestyksessä tiedot: annettu komento ja UTC-aika, leveyspiiri, pituuspiiri, horisontaalinen tarkkuus, korkeus, koordinaatin korjauksen tyyppi, kohteen suunta, kohteen nopeus (km/h), kohteen nopeus (solmuina), päivämäärä ja käytettävien satelliittien määrä. Jotta vastauksesta saataisiin käyttöön tarvittavat koordinaatit, se pitää vielä pilkkoa. Vastauksen pilkkomiseen käytetään PHP:n split()- ja list()-funktioita. /29, s.424–425/

```
list($alkujakello, $latitude, $longitude, $hdop, $altitude,
$fix, $cog, $spkm, $spkn, $pvm, $nsat) = split('[,]', $tulos);
```

Muuttuja ”$tulos” on siis laitteen antama vastaus kokonaisuudessaan ja se pilkotaan yllä näkyviin muuttujiin pilkun ollessa kahden eri tiedon erottajana. Näin saadaan käyttöön halutut arvot, jotka sisältyvät nyt muuttujiin ”$latitude” ja ”$longitude”. Muilla saaduilla muuttujilla ei tässä tarkoituksessa tehdä mitään, joten ne vaan jätetään käyttämättä.

Muuttujien arvoja ei voida kuitenkaan suoraan syöttää koordinaateiksi karttaan, sillä laite antaa koordinaatit eri muodossa, kuin Google Maps API ne hyväksyy. Laitteen antamista koordinaateista leveyspiirit ovat muodossa ”ddmm.mmmmN/S”, jossa ”dd” tarkoittaa asteita (00–90), ”mm.mmmm” tarkoittaa minuutteja (00.0000–59.9999) ja ”N/S” määräytyy sen mukaan, onko kysymyksessä eteläinen vai pohjoinen leveys. Pituuspiirit ovat puolestaan muodossa ”dddmm.mmmmE/W”, jossa ”ddd” tarkoittaa asteita (000–180), ”mm.mmmm” tarkoittaa minuutteja (00.0000–59.9999) ja ”E/W” määräyryy sen mukaan, onko kysymyksessä itäinen vai läntinen pituus. /29, s.425/

Koordinaatit pitää muuttaa Google Maps API:n hyväksymään muotoon, joka on dd.dddddd, eli asteiden desimaalimuotoon /30/. Muuntamista varten muuttujiin tallennetut koordinaatit pitää siis pilkkoa edelleen pienempiin osiin. Pilkkomiseen käytetään tällä kertaa substr()-funktiota, jolla muuttuja saadaan pilkottua merkkitasolla.

```
$lat_1 = substr("$latitude", 0, -8);
$lat_2 = substr("$latitude", 2, -1);
$long_1 = substr("$longitude", 0, -8);
$long_2 = substr("$longitude", 3, -1);
$latit = ($lat_2 / 60) + $lat_1;
$longit = ($long_2 / 60) + $long_1;
```

Funktioilla erotellaan jo asteina oleva osa minuutteina olevasta osasta. Muuttujaan ”$lat_1” tallennetaan leveyspiiristä juuri asteosa, joka alkaa siis alusta (0) ja loppuu kahdeksan merkkiä taaksepäin arvon lopusta katsottuna (-8). Leveyspiirin minuuttiosa talletetaan puolestaan muuttujaan ”$lat_2”, mutta jätetään pois viimeinen

merkki ”N/S”. Pituuspiirien osat erotellaan samalla tavalla muuttujiin ”$long_1” ja ”$long_2”. Oikeassa muodossa oleva koordinaatti saadaan jakamalla minuuttiosa 60:llä ja lisäämällä siihen sen jälkeen valmis asteosa.

Google Maps API:ssa piirien eteläisyys ja läntisyys merkitään miinus-merkillä koordinaatin edessä, mutta koska sovellus on suunniteltu toimimaan tällä hetkellä pelkästään Suomessa ja Suomi sijaitsee maapallon koillisella neljänneksellä, jätetään etumerkin muuntaminen kokonaan käsittelemättä /30/. Maailmanlaajuisessa käytössä tulisi toki ottaa huomioon myös laitteelta tulleen koordinaatin jälkeinen kirjain (N/S/E/W) ja muuttaa valmis desimaaliarvo tarvittaessa negatiiviseksi.

Muokkaamisen jälkeen koordinaatit tulisi vielä saada javascript-kielellä kirjoitetun, upotettavan, kartan määrittelevän ohjelmakoodin sekaan. Tällä hetkellä PHP:n muuttujiin talletetut arvot pitäisi siis saada siirrettyä javascriptin omiin muuttujiin. Siirto saadaan toteutettua yksinkertaisella echo()-funktiolla.

```
<script type="text/javascript">
      var latitu = "<?php echo $latit; ?>";
      var longitu = "<?php echo $longit; ?>";
```

Koska PHP-koodin suoritus tapahtuu jo palvelimella ja javascript vasta selaimella, voidaan PHP-muuttujien arvot tulostaa javascript-muuttujien määrittelyihin jo ennen kuin javascript suoritetaan /31/. Tällä tavalla javascript-muuttujien ”latitu” ja ”longitu” arvot ovat samat kuin PHP-muuttujien ”$latit” ja ”$longit”.

```
function load() {
  if (GBrowserIsCompatible()) {
    var map = new GMap2(
    document.getElementById("map"));
    map.addControl(new GSmallMapControl());
    map.setCenter(
    new GLatLng(latitu, longitu), 15);
    function createMarker(point, text, title) {
      var marker = new GMarker(point,{title:title});
      GEvent.addListener(marker, "click", function() {
        marker.openInfoWindowHtml(text);
      });
```

```
        return marker;
      }
      var marker = createMarker(
      new GLatLng(latitu, longitu),
      'GPS-koordinaatit:<br /><?php echo $latitude; ?>
      <br /><?php echo $longitude; ?>', 'Sijainti');
      map.addOverlay(marker);
    }
  }
  </script>
```

Javascript-koodissa määritellään myös kartta sisältämään navigointinäppäimet, kartan keskittäminen saatuihin koordinaatteihin, tarkennustason oletusarvo 15 sekä sijainnin osoitinmerkki, jota klikkaamalla avataan tietoikkuna, jossa kerrotaan alkuperäiset laitteelta saadut koordinaatit.

Sivun <body>-osaan pitää enää vain määritellä, että aiemmin luotu javascript-koodi suoritetaan, kun sivu ladataan. Kartta asetetaan sivulla haluttuun paikkaan -tagin avulla, ja samalla määritellään kartan haluttu koko. Sen jälkeen karttasovellus on valmis toimimaan yhdessä laitteen kanssa.

```
<body onload="load()" onunload="GUnload()" >
<table>
 </tr>
  <td>
   <div id="map" style="width: 500px; height: 300px"></div>
  </td>
 </tr>
</table>
```

Kuva 6: Kartta, johon on haettu laite3:n sijainti.

5.6 Moduulin ja sovelluksen käyttömahdollisuudet

Moduulin ja sen etähallintasovelluksen käyttömahdollisuudet ovat periaatteessa rajattomat ja oikeastaan juuri etähallintamahdollisuus tekee siitä todella monipuolisesti muokattavan. Esimerkiksi tässäkin työssä mukana ollutta karttasovellusta voi jo soveltaa moneen käyttöön ja se on vain yksi mahdollinen sovellusalue. Pienen kokoisen moduulin voi laittaa kiinni lähes mihin tahansa, ja sen sijainnin voi nähdä, mistä tahansa tietokoneelta.

Kuljetusyritykset voisivat esimerkiksi seurata kuljetustensa kulkua tai omistaja voisi paikallistaa kadonneen lemmikkinsä. Ajoneuvojen vuokraajat voisivat tarvittaessa paikantaa vuokralla olevan ajoneuvon tai matkustaja jopa itsensä. Moduulin voisi myös yhdistää esimerkiksi lämpömittariin tai sensoriin, jolloin saisi tiedon kesämökin lämpötilasta tai siitä että mökin ovi on avattu. Kun hallinta pystytään tekemään etäältä, ei myöskään asetusten muutoksia tai päivityksiä tarvitse päästä tekemään laitteelle fyysisesti, vaan laite voi olla omassa paikassaan vaikka kiinteästi.

Laitteita voitaisiin myös myydä eri osapuolille palveluna, jolloin kaikille osapuolille olisi omat tunnukset sivustolle, jossa sallittaisiin vain omien laitteiden tarkastelu. Siten kaikkien laitteiden hallinnointi tapahtuisi samalta sivustolta. Olisi myös mahdollista sallia useamman laitteen hallinnointi kerralla, jolloin esimerkiksi päivitykset ja asetusten muutokset saataisiin tehtyä kerralla kaikkiin yrityksen laitteisiin.

LÄHTEET

1. Wikipedia. PHP [verkkodokumentti]. Saatavissa: http://fi.wikipedia.org/wiki/PHP.

2. TIOBE Software. TIOBE Programming Community Index for April 2009 [verkkodokumentti]. 2009 [viitattu 28.4.2009]. Saatavissa: http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html.

3. PHP Group. PHP: Types [verkkodokumentti]. Saatavissa: http://www.php.net/manual/fi/language.types.php.

4. Wikipedia. MySQL [verkkodokumentti]. Saatavissa: http://fi.wikipedia.org/wiki/MySQL.

5. Kokkarinen, I. Java, Prolog ja Python: Tehokas näkökulma ohjelmointiin. 1.painos. Helsinki: IT Press, 2004. 410 s.

6. PHP Group. PHP: md5 – Manual [verkkodokumentti]. Saatavissa: http://fi2.php.net/manual/en/function.md5.php.

7. PHP Group. PHP: sha1 – Manual [verkkodokumentti]. Saatavissa: http://fi2.php.net/manual/en/function.sha1.php.

8. PHP Group. PHP: crc32 – Manual [verkkodokumentti]. Saatavissa: http://fi2.php.net/manual/en/function.crc32.php.

9. RFC 1321. The MD5 Message-Digest Algorithm. IETF, 1992. 21 s. Saatavissa: http://tools.ietf.org/html/rfc1321.

10. Wikipedia. MD5 [verkkodokumentti]. Saatavissa: http://fi.wikipedia.org/wiki/MD5.

11. Google. Google Maps API [verkkodokumentti]. Saatavissa: http://code.google.com/intl/fi/apis/maps/.

12. Telit. GE863-GPS [verkkodokumentti]. Saatavissa: http://www.telit.com/en/products.php?p_id=3&p_ac=show&p=8.

13. Wikipedia. Subscriber Identity Module [verkkodokumentti]. Saatavissa: http://fi.wikipedia.org/wiki/SIM.

14. Penttinen, J. Tietoliikennetekniikka. 1.painos. Helsinki: WSOY, 2006. 234 s.

15. Wikipedia. NMEA 0183 [verkkodokumentti]. Saatavissa: http://en.wikipedia.org/wiki/NMEA.

16. DePriest, D. NMEA data [verkkodokumentti]. Saatavissa: http://www.gpsinformation.org/dale/nmea.htm.

17. Frank Durda. The AT Command Set Reference – History [verkkodokumentti]. 2004. Saatavissa: http://nemesis.lonestar.org/reference/telecom/modems/at/history.html.

18. Wikipedia. Hayes Microcomputer Products [verkkodokumentti]. Saatavissa: http://en.wikipedia.org/wiki/Hayes_Microcomputer_Products.

19. Wikipedia. Hayes Command Set [verkkodokumentti]. Saatavissa: http://en.wikipedia.org/wiki/Hayes_command_set.

20. CellularOnline. Hayes AT Commands [verkkodokumentti]. Saatavissa: http://www.cellular.co.za/hayesat.htm.

21. Miettinen, S. GPS Käsikirja. 3.painos. Helsinki: Genimap, 2006. 192 s.

22. Poutanen, M. GPS-Paikanmääritys. Hämeenlinna: Ursa, 1998. 269 s.

23. ESF. Natural Resources Information Systems [verkkodokumentti]. Saatavissa: http://www.esf.edu/for/bevilacqua/for324/hw.htm.

24. United States Airforce. Globla Positioning System [verkkodokumentti]. 2007. Saatavissa: http://www.af.mil/factsheets/factsheet.asp?id=119.

25. Concord-Carlisle Highschool. GPS: the Mathematics and the Technology. [verkkodokumentti]. Saatavissa: http://mail.colonial.net/~abeckwith/gps.html.

26. Sun Microsystems. Changes in release 3.23.49 (14 February 2002) [verkkodokumentti]. Saatavissa: http://dev.mysql.com/doc/refman/4.1/en/news-3-23-49.html.

27. Wikipedia. MyISAM [verkkodokumentti]. Saatavissa: http://en.wikipedia.org/wiki/MyISAM.

28. Google. Google Maps API Concepts [verkkodokumentti]. Saatavissa: http://code.google.com/intl/fi-FI/apis/maps/documentation/index.html#SpecifyingSensor.

29. Telit. AT Commands Reference Guide [verkkodokumentti]. Saatavissa: http://www.telit.com/module/infopool/download.php?id=542.

30. Google. Google Maps API Reference [verkkodokumentti]. Saatavissa: http://code.google.com/intl/fi-FI/apis/maps/documentation/reference.html#GLatLng

31. PHP Group. PHP: Introduction [verkkodokumentti]. Saatavissa: http://www.php.net/manual/fi/introduction.php.